Amendments to the Claims:

Listing of Claims:

Claim 1 (currently amended):

A program flow method in a program component system, the program component system comprising a running time system and several components each having a program portion, said method comprising the steps of:

a) triggering a call of a called first component of said several components by call information disposed in a docking point of a calling second component of said several components and

the following steps that are performed after [[a]] said called first component ~~of said several components~~ has been called and during the execution of the program portion of said called first component:

[[a]] b) acquiring for said called first component, by means of the running time system, first data from said [[a]] calling second component ~~of said several components~~ without any need for programmer-defined interfaces in said calling second component, wherein the source of the first data to be acquired within said calling second component is selected according to a definition of said called first component; and

[[b]] c) disposing, from said called first component, by means of the running time system, second data into said calling second component without any need for programmer-defined interfaces in said calling second component, wherein a target to which said second data is ~~to be~~ deposited within said calling second component is selected according to a definition of the called first component.

Claim 2 (currently amended):

The method according to claim 1, characterized in that the first data transmitted during said acquiring are transferred from a memory image portion of said calling second component into a transfer data region of said called first component, and/or that the second data transmitted during said disposing are transferred from a transfer data region of said called first component into a memory image portion of said calling second component.

Claim 3 (currently amended):

The method according to claim 1, characterized in that said acquiring and/or said disposing is carried out without the cooperation of said calling second component.

Claim 4 (currently amended):

The method according to claim 1, characterized in that said calling second component is inactive during said acquiring and/or said disposing.

Claim 5 (currently amended):

The method according to claim 2, characterized in that said transfer data region of said calling second component is located in a saving region during said acquiring and said disposing.

Claim 6 (currently amended):

The method according to claim 1, characterized in that local and/or non-persistent data of said calling second component are transmitted during said acquiring and/or said disposing.

Claim 7 (currently amended):

The method according to claim 1, characterized in that during the execution of the program portion of said called first component a waiting list is made indicating which of the data of said called first component require disposing.

Claim 8 (previously presented):

The method according to claim 1, characterized in that a called component can directly access an access data region comprising the data fields defined and /or available in a calling component.

Claim 9. Canceled.

Claim 10 (previously presented):

A method of expanding a program component system comprising several components by a further component, said method comprising the steps of:

a) searching for docking points for said further component in said program component system, which docking points correspond to an inheritance parameter determined by a definition of said further component; and

b) modifying each of said several components of the program component system in which at least one docking point was found by entering call information at each docking point found, said call information indicating said further component, wherein said expansion of said program component system is completed without any need for programmer-defined expansion interfaces in said several components.

Claim 11 (previously presented):

The method according to claim 10, characterized in that all interaction interfaces of said several components are predefined as potential docking points.

Claim 12 (previously presented):

The method according to claim 10, characterized in that all interaction screen fields referenced by said several components and/or all print mask output fields and/or all access operations on persistent data are predefined as potential docking points.

Claim 13 (previously presented):

The method according to claim 10, characterized in that said entering call information at each docking point a call of the further component from each of said several components into which said call information was entered.

Claim 14 (previously presented):

The method according to claim 10, characterized by additionally:

c) generating at least one binary object from the definition of the further component.

Claim 15 (original):

The method according to claim 14, characterized in that a maximum of one binary object is generated for each docking point that has been found.

Claim 16 (previously presented):

The method according to claim 15, characterized in that while generating each binary object, the memory allocation is considered in the one component of the program component system which includes the underlying docking point.